**CSE 323**

**OS Project Report**

**Faculty: Nova Ahmed**

**Section: 1**

**Members: ID:**

**MD. Ali Zubair 1210374042**

**MD. Rahber-e-Alam Raafa 1210953042**

**Syed Ishtiaque Ahmad Jitu 1210367042**

**Project Name:**

Temperature and Humidity Monitor and Detection device.

**Project Objective:**

To measure the surrounding temperature and humidity levels in the atmosphere and return the calibrated values back to the user for recording and analysis.

**Equipments Used:**

1. Raspberry Pi.
2. DHT11 Temperature and Humidity Sensor.
3. 8GB Macro SD card.
4. 10 kilo ohm resistor.
5. Bread Board.
6. Connecting Wires.
7. Cell Phone Charger (Power Adapter).
8. HDMI to VGA adaptor.
9. Display Unit (Monitor/Projector).

**Equipment Set-Up:**

Load the memory card onto the raspberry pi and connect the charger. Install the “Raspbian” operating system and connect the DHT11 sensor to the raspberry pi’s GPIO ports using the bread board.



Figure . DHT11 Sensor

**RPi VCC (pin 1) -> DHT11 pin 1**

**RPi GPIO4 (pin 7) -> DHT11 pin 2**

**RPi GND (pin 6) -> DHT11 pin 4**

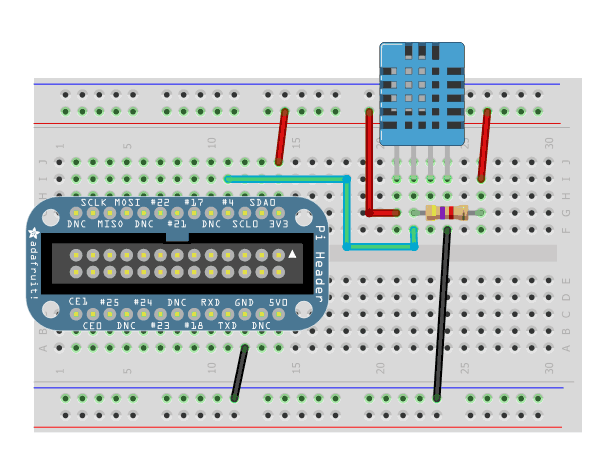


Figure . Pin Configuration of DHT11 Sensor

The connection is given using 3 wires, the left pin is connected to VCC (pin 1), the data pin is next after VCC and is connected to pin 7. Next pin is NC (no connection). Finally the last pin is connected to GND(pin 25).To prevent random data generation, we connect a 10K resistor between data and VCC pin of DHT11.The 10 kilo ohm resistor initiates a potential difference and ultimately, a current flow across the circuit.

The DHT11 uses its own serial interface, which can be interrogated using the wiringPi C library. WiringPi which uses C like Arduino language is used to read the sensor value.

WiringPi installation command:

sudo apt-get install git-core build-essential

git clone git://git.drogon.net/wiringPi

cd wiringPi

./build

Raspberry Pi initiates the data transmission process by pulling the data bus low for about 18ms and keeps it HIGH for about 20-40μs before releasing it. Subsequently, the sensor responds to the Pi's data transfer request by pulling the data bus LOW for 80μs followed by 80μs of HIGH. At this point Pi is ready to receive data from the sensor. Data is sent in packet of 40 bits (5 bytes) via the data line with the most significant bit at the beginning.

Data is transmitted in the following order:- Integer Part of Relative Humidity--->Decimal Part of Relative Humidity--->Integer Part of Temperature--->Decimal Part of Temperature---> Checksum. Checksum consists the last 8 bits of each part. Transmission of '0' & '1' is done by varying the width of the pulse. For transmitting '0' the data bus is held HIGH for 26-28μs, and 70μs for transmitting '1'.

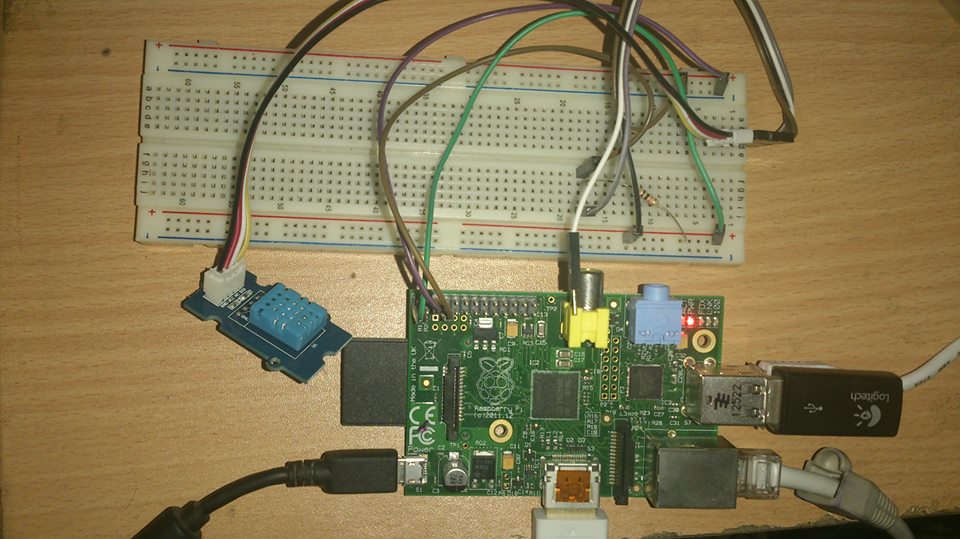
****

Figure . DHT11 Sensor fully configured with Raspberry Pi.

**C-Code to Collect the Temperature and Humidity Data:**

The following C code reads the surrounding current temperature and humidity from the sensor every 5 seconds and displays the reading on the screen, simultaneously creating a log file named “Data.txt”, where the data are recorded along with the time and date of the day.

#**include** <wiringPi.h>

#**include** <stdio.h>

#**include** <stdlib.h>

#**include** <stdint.h>

#**include** <time.h>

#**define** MAX\_TIME **85**

#**define** DHT11PIN **7**

#**define** ATTEMPTS **5**

**int** dht11\_val[**5**]={**0**,**0**,**0**,**0**,**0**};

**int** dht11\_read\_val()

{

uint8\_t lststate=HIGH;

uint8\_t counter=**0**;

uint8\_t j=**0**,i;

**for**(i=**0**;i<**5**;i++)

dht11\_val[i]=**0**;

**pinMode**(DHT11PIN,OUTPUT);

**digitalWrite**(DHT11PIN,LOW);

**delay**(**18**);

**digitalWrite**(DHT11PIN,HIGH);

**delayMicroseconds**(**40**);

**pinMode**(DHT11PIN,INPUT);

**for**(i=**0**;i<MAX\_TIME;i++)

{

counter=**0**;

**while**(**digitalRead**(DHT11PIN)==lststate)

{ counter++;

**delayMicroseconds**(**1**);

**if**(counter==**255**)

**break**;

}

lststate=**digitalRead**(DHT11PIN);

**if**(counter==**255**)

**break**;

// top 3 transitions are ignored

**if**((i>=**4**)&&(i%**2**==**0**)){

dht11\_val[j/**8**]<<=**1**;

**if**(counter>**16**)

dht11\_val[j/**8**]|=**1**;

j++;

}

}

// verify checksum and print the verified data

**if**((j>=**40**)&&(dht11\_val[**4**]==((dht11\_val[**0**]+dht11\_val[**1**]+dht11\_val[**2**]+dht11\_val[**3**])& **0xFF**)))

{

time\_t result = **time**(**NULL**);

**printf**("Relative Humidity: **%d**.**%d**,Temperature: **%d**.**%d** on **%s**",dht11\_val[**0**],dht11\_val[**1**],dht11\_val[**2**],dht11\_val[**3**],**asctime**(**localtime**(&result)));

FILE \*fp;

fp = **fopen**("Data.txt", "a");

**if**(fp==**NULL**)

{

**printf**("File could not be opened**\n**");

}

**else**

{

**fprintf**(fp, "**%d**.**%d** **%d**.**%d** **%s**",dht11\_val[**0**],dht11\_val[**1**],dht11\_val[**2**],dht11\_val[**3**],**asctime**(**localtime**(&result)));

**fclose**(fp);

}

**return** **1**;

}

**else**

**return** **0**;

}

**int** main(**void**)

{

**int** a = **0**;

**int** attempts=ATTEMPTS;

**if**(**wiringPiSetup**()==-**1**)

**exit**(**1**);

**for**(a=**0**; a<**5**; )

{

**int** success = **dht11\_read\_val**();

**if** (success) {

a++;

**delay**(**5000**);

}

}

**return** **0**;

}

This code is compiled and executed from the TxTerminal (Raspbian CMD equivalent) by the following commands:

gcc -o dht11 dht11.c -L/usr/local/lib -lwiringPi

sudo ./dht11

**Java-Code to display the results onto a bar chart:**

The following Java code reads the temperature and humidity log from the “Data.txt” file and displays the data on the screen as well as creates a “Temperature and Humidity vs. Time” bar chart for easier analysis.

**import** java.awt.Color;

**import** java.awt.Dimension;

**import** java.awt.GradientPaint;

**import** java.io.\*;

**import** java.util.\*;

**import** org.jfree.chart.ChartFactory;

**import** org.jfree.chart.ChartPanel;

**import** org.jfree.chart.JFreeChart;

**import** org.jfree.chart.axis.CategoryAxis;

**import** org.jfree.chart.axis.CategoryLabelPositions;

**import** org.jfree.chart.axis.NumberAxis;

**import** org.jfree.chart.plot.CategoryPlot;

**import** org.jfree.chart.plot.PlotOrientation;

**import** org.jfree.chart.renderer.category.BarRenderer;

**import** org.jfree.data.category.CategoryDataset;

**import** org.jfree.data.category.DefaultCategoryDataset;

**import** org.jfree.ui.ApplicationFrame;

**import** org.jfree.ui.RefineryUtilities;

public class **BarChartDemo** extends **ApplicationFrame** {

public static **int** count = **0**;

public static **String** time[] = **new** **String**[**100**];

public static **String** humidity[] = **new** **String**[**100**];

public static **String** temp[] = **new** **String**[**100**];

public BarChartDemo(final **String** title) {

**super**(title);

final **CategoryDataset** dataset = createDataset();

final **JFreeChart** chart = createChart(dataset);

final **ChartPanel** chartPanel = **new** **ChartPanel**(chart);

chartPanel.setPreferredSize(**new** **Dimension**(**500**, **270**));

setContentPane(chartPanel);

}

private **CategoryDataset** createDataset() {

final **String** series1 = "Temperature";

final **String** series2 = "Humidity";

final **DefaultCategoryDataset** dataset = **new** **DefaultCategoryDataset**();

**for**(**int** i=**0**;i<count;i++)

{

**double** valueTemp=**Double**.parseDouble(temp[i]);

dataset.addValue(valueTemp, series1, time[i]);

**double** valueHumid=**Double**.parseDouble(humidity[i]);

dataset.addValue(valueHumid, series2, time[i]);

}

**return** dataset;

}

private **JFreeChart** createChart(final **CategoryDataset** dataset) {

// create the chart...

final **JFreeChart** chart = **ChartFactory**.createBarChart(

"Temperature & Humidity Bar Chart", // chart title

"Time", // domain axis label

"Value", // range axis label

dataset, // data

**PlotOrientation**.**VERTICAL**, // orientation

**true**, // include legend

**true**, // tooltips?

**false** // URLs?

);

chart.setBackgroundPaint(**Color**.white);

final **CategoryPlot** plot = chart.getCategoryPlot();

plot.setBackgroundPaint(**Color**.lightGray);

plot.setDomainGridlinePaint(**Color**.white);

plot.setRangeGridlinePaint(**Color**.white);

// set the range axis to display integers only...

final **NumberAxis** rangeAxis = (**NumberAxis**) plot.getRangeAxis();

rangeAxis.setStandardTickUnits(**NumberAxis**.createIntegerTickUnits());

// disable bar outlines...

final **BarRenderer** renderer = (**BarRenderer**) plot.getRenderer();

renderer.setDrawBarOutline(**false**);

// set up gradient paints for series...

final **GradientPaint** gp0 = **new** **GradientPaint**(

**0.0f**, **0.0f**, **Color**.blue,

**0.0f**, **0.0f**, **Color**.lightGray

);

final **GradientPaint** gp1 = **new** **GradientPaint**(

**0.0f**, **0.0f**, **Color**.green,

**0.0f**, **0.0f**, **Color**.lightGray

);

final **GradientPaint** gp2 = **new** **GradientPaint**(

**0.0f**, **0.0f**, **Color**.red,

**0.0f**, **0.0f**, **Color**.lightGray

);

renderer.setSeriesPaint(**0**, gp0);

renderer.setSeriesPaint(**1**, gp1);

renderer.setSeriesPaint(**2**, gp2);

final **CategoryAxis** domainAxis = plot.getDomainAxis();

domainAxis.setCategoryLabelPositions(

**CategoryLabelPositions**.createUpRotationLabelPositions(**Math**.**PI** / **6.0**)

);

**return** chart;

}

public **void** fileRead(**String** fileName){

**try**{

**Scanner** scan = **new** **Scanner**(**new** **File**(fileName));

**while**(scan.hasNextLine()){

humidity[count] = scan.next();

temp[count] = scan.next();

scan.next();scan.next();scan.next();

time[count] = scan.next();

scan.next();

count++;

}

scan.close();

}

**catch**(**Exception** e){

e.printStackTrace();

}

}

public static **void** main(final **String**[] args) {

**BarChartDemo** dr = **new** **BarChartDemo**(**null**);

dr.fileRead("F:/javaPrograms/BarChart/Data.txt");

**System**.out.println("Time" + " " + "Humidity" + " " + "Temperature");

**for**(**int** i=**0**; i<count; i++){

**System**.out.println(time[i] + " " + humidity[i] + " " + temp[i]);

}

final **BarChartDemo** demo = **new** **BarChartDemo**("Bar Chart Demo");

demo.pack();

**RefineryUtilities**.centerFrameOnScreen(demo);

demo.setVisible(**true**);

}

}

The above code is compiled and executed by the following command:

javac -classpath .:classes:/home/pi/Desktop/lib/’\*’ BarChartDemo.java

sudo java -classpath .:classes:/home/pi/Desktop/lib/’\*’ BarChartDemo

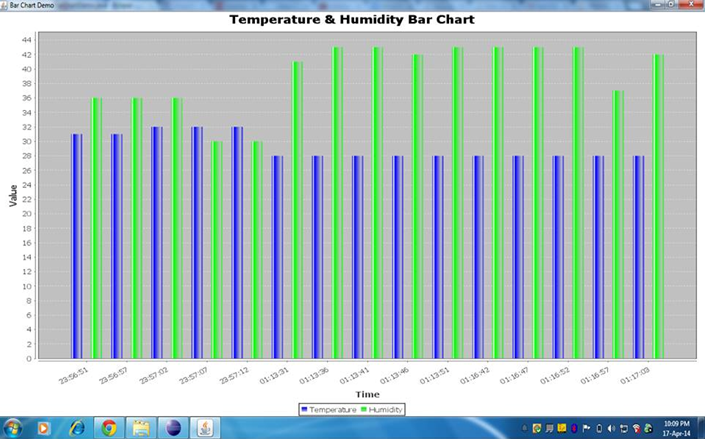
****

Figure 4. “Temperature and Humidity vs. Time” Bar Chart.

**References:**

<https://www.youtube.com/watch?v=cxR84q5cbUc>

<https://learn.adafruit.com/dht/overview>